AFRL-RI-RS-TR-2013-060

DESIGN FOR A MANUFACTURING METHOD FOR MEMRISTOR-BASED NEUROMORPHIC COMPUTING PROCESSORS

UNIVERSITY OF PITTSBURGH

MARCH 2013

FINAL TECHNICAL REPORT

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

AFRL-RI-RS-TR-2013-060 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/ S /

NATHAN McDONALD
Work Unit Manager

/ S /

JOSEPH A. CAROLI
Acting Technical Advisor, Computing &
Communications Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* MARCH 2013	**2. REPORT TYPE** FINAL TECHNICAL REPORT	**3. DATES COVERED** *(From - To)* SEP 2011 – SEP 2012

4. TITLE AND SUBTITLE DESIGN FOR A MANUFACTURING METHOD FOR MEMRISTOR-BASED NEUROMORPHIC COMPUTING PROCESSORS	**5a. CONTRACT NUMBER** FA8750-11-1-0271
	5b. GRANT NUMBER N/A
	5c. PROGRAM ELEMENT NUMBER 62788F
6. AUTHOR(S) Yiran Chen and Beiye Liu	**5d. PROJECT NUMBER** T2NC
	5e. TASK NUMBER PI
	5f. WORK UNIT NUMBER TT
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University Of Plattsburgh Electrical and Computer Engineering Department 4200 Fifth Ave Pittsburgh, PA 15213	**8. PERFORMING ORGANIZATION REPORT NUMBER** N/A
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory/RITB 525 Brooks Road Rome NY 13441-4505	**10. SPONSOR/MONITOR'S ACRONYM(S)** N/A
	11. SPONSORING/MONITORING AGENCY REPORT NUMBER AFRL-RI-RS-TR-2013-060

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. PA# 88ABW-2013-1074
Date Cleared: 5 March 2013

13. SUPPLEMENTARY NOTES

14. ABSTRACT
It is well received that conventional CMOS technology is approaching its physical limitations. Researchers have started to explore the potential replacement by leveraging the advances of nanotechnology. Very recently, memristor attracted growing attentions since the first physical realization reported by HP Labs in 2008. Unique characteristics like non-volatility, re-configurability, and analog state storage make memristor become a very promising candidate for the realization of artificial neural systems. In this project, we developed a SPICE-compatible model of memristor and designed CMOS-mimicked memristor cells for system development. Then we proposed a memristor-based design of bidirectional transmission excitation/inhibition synapses and implemented a neuromorphic computing system based on our proposed synapse designs. The robustness of our system is also evaluated by considering the actual manufacturing variability with the emphasis on process variations. After that, we discussed memristor-based crossbar neuromorphic architecture. Finally, we compared the designs of synapse network-based and crossbar-based neuromorphic computing systems.

15. SUBJECT TERMS
Memristor, Neuromorphic, Nanotechnology, CMOS Technology

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			NATHAN MCDONALD
U	U	U	UU	34	**19b. TELEPONE NUMBER (*Include area code*)** N/A

TABLE OF CONTENTS

Section **Page**

LIST OF FIGURES

LIST OF TABLES

1.0 SUMMARY

It is well received that conventional CMOS technology is approaching its physical limitations. Researchers have started to explore the potential replacement by leveraging the advances of nanotechnology. Very recently, memristors attracted growing attentions since the first physical realization was reported by Hewlett-Packard (HP) Laboratories in 2008. Unique characteristics like non-volatility, reconfigurability, and analog state storage made memristors a very promising candidate for the realization of artificial neural systems. In this project, we developed a SPICE-compatible model of a memristor and designed CMOS-mimicked memristor cells for system development. Then we proposed a memristor-based design of bidirectional transmission excitation/inhibition synapses and implemented a neuromorphic computing system based on our proposed synapse designs. The robustness of our system is also evaluated by considering the actual manufacturing variability with the emphasis on process variations. Next, we discussed memristor-based crossbar neuromorphic architecture. Finally, we compared the designs of synapse network-based and crossbar-based neuromorphic computing systems.

2.0 INTRODUCTION

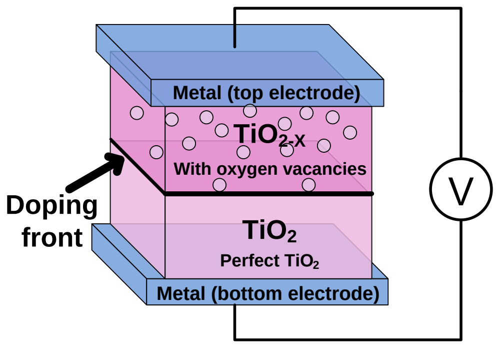
Although the memristor was predicted as the 4th fundamental circuit element in 1971 by Professor Chua [1], the first realization of a physical memristive system was reported by HP Labs in 2008 [2]. The memristive effect is shown as a pinched hysteresis I-V curve, which becomes the basis for resistive memories [3]. Memristance can be theoretically described as the relationship between the magnetic flux φ and the electric charge q through the device as [1]

$$\mathrm{d}\varphi = M \cdot \mathrm{d}q. \tag{1}$$

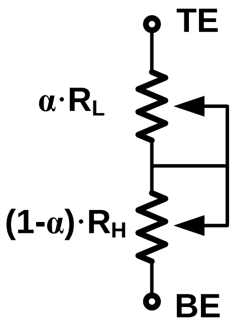
Figure 1(a) shows the conceptual structure of a TiO_2 memristor [2][4]. A perfect TiO_2 layer acts as an insulator or highly resistive conductor, while the conductivity of the oxygen-deficient titanium dioxide TiO_{2-x} layer is much higher. The resistance of the entire memristive system, or memristance, can be controlled by moving the doping boundary between the TiO_2 and TiO_{2-x} regions. As shown in Figure 1(b), the overall memristance can be calculated as

$$M(\alpha) = \alpha \cdot R_l + (l - \alpha) \cdot R_h,. \tag{2}$$

where R_h and R_l represent the conductivities per length of TiO_2 and TiO_{2-x}, respectively.



(a) TiO_2 Memristor Stucture



(b) Equivalent Circuit

Figure 1. TiO_2 thin-film memristor [4]

In general, memristors have the following unique properties that make them become very promising devices for artificial neural system implementation. 1) Memristance relies on the history of the total electric charge flowing through the device [1][5]. 2) Memristors are non-volatile [6][7], that is, the memristance (resistance) of the device can be retained even after the system is powered off. No leakage or refresh power overheads are introduced into during the information storage. 3) Memristors can be used as an analog device of which the resistance state can be programmed continuously.

3.0 METHODS, ASSUMPTIONS, AND PROCEDURES

3.1. Memristor Model

We developed a SPICE-compatible compact model of TiO_2-TiO_{2-x} memristors based on classic ion transportation theory. Our model is shown to simulate important dynamic memristive properties, e.g., the doping front motion and real-time memristance switching, which are critical in memristor-based analog circuit designs. The model, as well as its analytical approximation, is validated with the experimentally obtained data from real devices. We divide a TiO_2-TiO_{2-x} memristor into three regions, namely, the conductive region, the transition region, and the insulating region, as shown in Figure 2. We also use w, λ, and D to denote the lengths of the conductive region, the transition region, and the entire device, respectively [1].

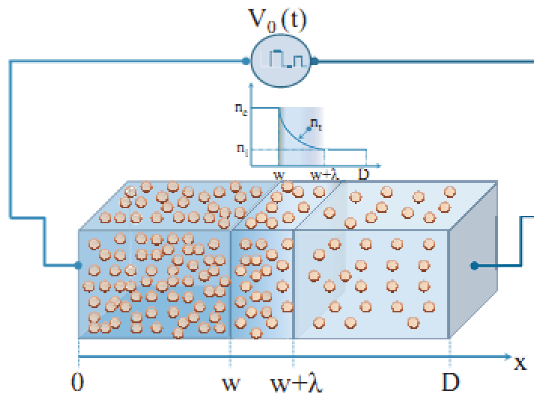


Figure 2. Region partitioning of *TiO₂-TiO₂₋ₓ* memristor

The real-time current density at position x in the memristor at time t can be expressed as

$$\boldsymbol{J_s(x,t) = n_s(x,t)q\mu E_s(x,t) - qD_q\frac{\mathrm{d}n_s(x,t)}{\mathrm{d}x}}. \tag{3}$$

Here the subscript $s = (c, t, i)$ denotes the parameter of conductive region, transition region, and insulating region, respectively. E_s and n_s represent the electric field and the electron density, respectively. μ is the mobility coefficient. q is the elementary charge. D_q is the diffusion constant. Eq. (3) shows that the current is generated mainly from the electron drifting in the electric field and the electron density gradient. D_q is typically small, so the second term in Eq. (3) is ignored in our model. If we ignore the variation of the cross section area and assume the current density is uniform in the memristor, i.e., $J_c = J_t = J_i$, then the relationships between the electric fields and electron densities of the three regions can be summarized as

$$n_c q\mu E_c(t) = n_t(x,t)q\mu E_t(x,t) = n_i(t)q\mu E_i(t). \tag{4}$$

Here we assume the E_s and n_s are uniform in the conductive and insulating regions and are determined only by the real-time applied voltage (except that n_c is a constant). In general, the evolution of the electron density in the insulating region under a time-varying electric field E_i can be calculated by

$$\frac{\mathrm{d}n_i(t)}{\mathrm{d}t} = \gamma_i\big(n_c - n_i(t)\big)E_i(t), \tag{5}$$

where γ_i is the electron generating coefficient in the insulating region. At a given time t and position x, the electric field E_t can be viewed as a linear function bounded by E_c and E_i,

$$E_t(x,t) = \frac{\big(E_i(t) - E_c(t)\big)}{\lambda}(x-\omega) + E_c(t). \tag{6}$$

We note that the voltage $V(t)$ applied on the two ends of the memristor equals the integral of the electric field along the device,

$$V(t) = E_c(t)\omega + \int_{\omega}^{\omega+\lambda} E_t(x,t)dx + E_i(t)(D-\lambda-\omega). \tag{7}$$

Combing Eq. (6) and (7), we have

$$E_c(t) = \frac{V(t)}{\omega + \frac{1}{2}\lambda + \frac{n_c}{n_i(t)}(D-\omega-\frac{1}{2}\lambda)}. \tag{8}$$

Finally, the transition region length λ reduces when the applied voltage $V(t)$ increases, which is approximated by

$$\lambda = \lambda_0 e^{-|V(t)|}. \tag{9}$$

In our model, λ_0 is the transition region length at $V(t) = 0$.

Without loss of generality, we assume the doping front starts moving from the left end of the device (x = 0) at time t = 0. $n_t(x,t)$ is the electron density at the position x in the transition region at time t. Compared to the electron density n_c at the boundary between the conductive region and the transition region, the change of the electron density at position x is $n_c - n_t(x,t)$, which is mainly due to the oxygen ion vacancy redistribution [6].

Because of the drifting of the doping front, the electron density increment in an infinitesimal time interval dt equals the difference between the electron densities at positions $(x - \mathrm{d}x)$ and x,

$$\begin{aligned} n_t(x-\mathrm{d}x,t) - n_t(x,t) &= \frac{\mathrm{d}n_t(x,t)}{\mathrm{d}x}(-\mathrm{d}x) = -\frac{\mathrm{d}n_t(x,t)}{\mathrm{d}x}v(x,t)\mathrm{d}t \\ &= \gamma_t\big(n_c - n_t(x,t)\big)E_t(x,t)\mathrm{d}t. \end{aligned} \tag{10}$$

Here γ_t is the electron generating coefficient in the transition region. Note that the maximum attainable electron density in the memristor device is n_c, or the region is fully conductive. Based on Eq. (10), the doping front velocity at position x can be calculated by

$$v(x,t) = -\frac{\gamma_t\big(n_c - n_t(x,t)\big)E_t(x,t)}{\frac{\mathrm{d}n_t(x,t)}{\mathrm{d}x}}, \tag{11}$$

where $\frac{\mathrm{d}n_t(x,t)}{\mathrm{d}x}$ can be derived from Eq. (4) and (6).

The motion of the transition region can be described by the average doping front moving velocity, which is defined as

$$\overline{v(t)} = \frac{\int_{\omega}^{\omega+\lambda} v(x,t)\mathrm{d}x}{\lambda}. \tag{12}$$

Here we simplify the expressions of n_c, $n_t(x,t)$, and $n_i(t)$ as n_c, n_t, and n_i, which are still the functions of *x* and/or *t*.

Substituting Eq. (4) and (6) into Eq. (11), we have

$$v(x,t) = \left(\frac{n_c}{n_t} - 1\right) n_c E_c(t) \cdot \frac{\gamma_t}{n_t^2} \cdot \frac{\lambda n_c n_i}{n_c - n_i}. \tag{13}$$

Substituting Eq. (4), (6), (8), and (11) into (12), the transition region moving velocity can be approximated by

$$\frac{\mathrm{d}w(t)}{\mathrm{d}t} \approx \bar{v}(t) = \gamma_t \lambda \beta \cdot \frac{1}{\frac{n_c}{n_i} - 1} \cdot \frac{V(t)}{\omega\left(1 - \frac{n_c}{n_i}\right) + \frac{1}{2}\lambda + \frac{n_c}{n_i}\left(D - \frac{1}{2}\lambda\right)}, \tag{14}$$

where $\beta = \frac{1}{4}\left(\frac{n_c}{n_i} - 1\right)^3 + \frac{2}{3}\left(\frac{n_c}{n_i} - 1\right)^2 + \frac{1}{2}\left(\frac{n_c}{n_i} - 1\right)$.

The memristance of the memristor can then be calculated by

$$R(t) = \frac{V(t)}{J_c A} = \frac{V(t)}{n_c q \mu E_c(t) A}, \tag{15}$$

where *A* is the cross section area of the memristor. Eq. (14) and (15) describe the dynamic changes of memristor device structure and electrical property, respectively.

3.2. CMOS-Mimicked Memristor Cell

3.2.1 CMOS-mimicked memristor cell design. We created a schematic cell model of CMOS-mimicked memristor that was used in the subsequent SPICE simulations, as shown in

Figure 3. The behavior of the memristor was emulated by using a CMOS circuit, which enabled binary training operation. A corresponding circuit symbol was also created to enable graphic-based circuit design. CMOS mimicked memristor had five pins rather than the two pins of a two-terminal memristor device because a latch was included in the design. Besides the two voltages at the top, *vtop*, and the bottom, *vbot*, of the virtual memristor device, we also need a power supply pair, *vdd* and *gnd*, and an output, *out*, to supply the latch and record its output.

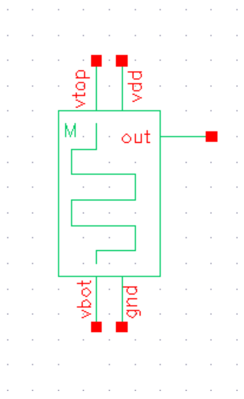


Figure 3. Symbol view of CMOS-mimicked memristor

The corresponding CMOS-mimicked memristor circuit schematic is shown in Figure 4. It was built up with an XOR gate, a latch, and an NMOS transistor. By controlling the gate voltage of the NMOS transistor, the resistance between the source and drain of the NMOS transistor was changed to emulate the memristance shift of a memristor. The difference between such a design and a real memristor was that our CMOS circuit could not fully reproduce the memristive behavior, which was determined by the historical effects of electronic excitation. The use of a latch allowed us to record the state of the "memristor" circuit and provide the proper response to the training operations.

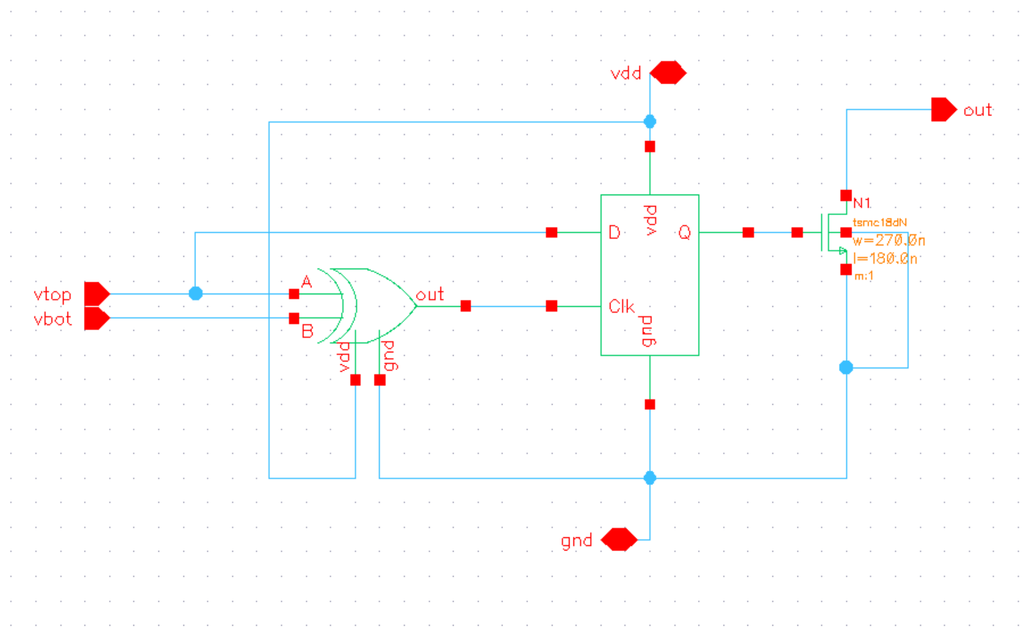


Figure 4. Schematic of CMOS-mimicked memristor

The truth table of the CMOS-mimicked memristor circuit is shown in Table 1. When the output *Q* of the latch was '1', the NMOS transistor functioned as a low resistance. When the output was '0', the NMOS transistor functioned as a high resistance. The XOR gate generated a CLK signal, which drove the latch. As shown in Table 1, the memristor only changed its resistance state when the value of *vtop* was different from the value of *vbot*. In fact, the state of the CMOS-mimicked memristor always changed to the direction indicated by the value of *vtop* during the state programming operation.

Table 1. CMOS-mimicked memristor truth table

V_top	V_bot	CLK	Q	Resistance
0	0	0	No change	No change
0	1	1	0	High
1	0	1	1	Low

The writing function is shown in Table 2. When *vtop* and *vbot* changed, a CLK signal was generated to trigger the corresponding training process. A latch circuit recorded the state of the CMOS-mimicked memristor and controlled the resistance of a NMOS transistor. When latch output Q was '1', the NMOS transistor was programmed to a low resistance state. Conversely, when Q was '0', the NMOS transistor was programmed to a high resistance state.

Table 2. CMOS-memristor based synapse operation table

Enable	Vout	Dtrain	Vtop	Vbot	CLK	Status
0	X	X	0	0	0	Operating
0	1	1	0	0	0	No training
1	0	1	1	0	1	R_H to R_L
1	0	1	0	1	1	R_L to R_H

The detailed writing circuit can be found in Figure 5. The whole synapse system can be divided into five major parts: latch, logic gates, write driver, pass gate, and CMOS-mimicked memristor.

Latch: Different from the latch in the CMOS-mimicked memristor circuit, this latch stored the value of *vout* when the enable signal *E* switched from '0' to '1' and held the value of *vout* for one clock cycle to produce the appropriate write signals.

Logic gates: These logic gates performed the necessary logic functions, including a 3-input OR gate and a 3-input NAND gate. They produced write signals based on three input signals (*E*, *vout*, and *Dtrain*) as shown in Table 2.

Write driver: The write driver was an inverter buffer that guaranteed sufficient strong write signals to drive the memristor.

Pass gate: Based on the enable signal *E*, the pass gate controlled the output signal *vout* by switching between *Z* and the value of the memristor when it was being trained.

Memristor: The memristor is the CMOS-mimicked memristor module.

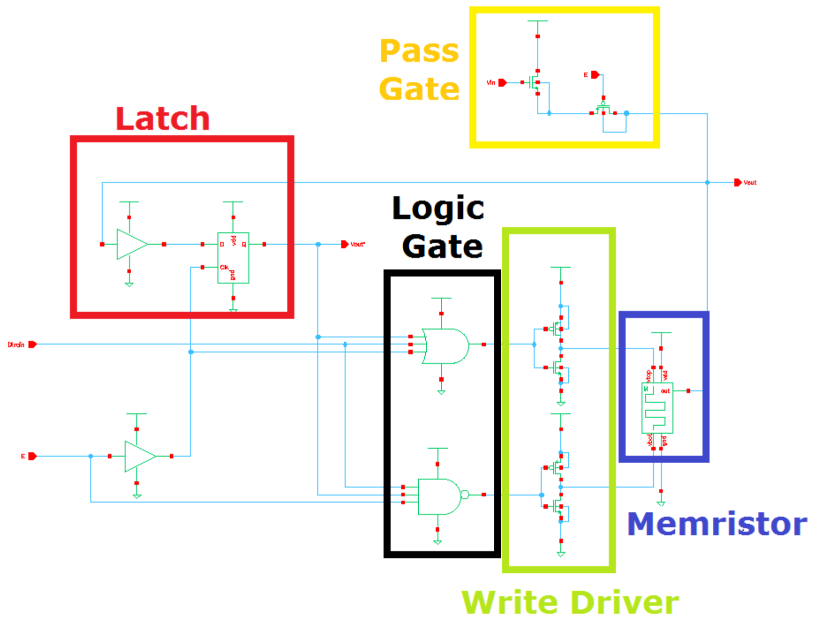


Figure 5. Schematic of the training circuit of CMOS-mimicked memristor

There are several differences between the designed CMOS-mimicked memristor circuit and the real memristor circuit. First, the inverters/buffers may not be necessary in the write drivers of the real memristor circuit. In the real memristor circuit, when the memristor is not being trained, the *vbot* should be set to '0', while the *vtop* should be floating. In the CMOS-mimicked circuit, both the *vtop* and the *vbot* have to be set to '0' when the mimicked memristor is being trained. The write driver is built up with two writing buffers, which generated *vtop* and *vbot* to adjust the resistance of the NMOS transistor.

3.2.2 CMOS-mimicked memristor cell layout. We also completed the layout of a single CMOS-mimicked memristor circuit, including the training circuit. As shown in Figure 6, the layout included five parts: Latch, Logic Gates, Write Drive, Pass Gate, and Memristor. There were 34 PMOS and 35 NMOS transistors in this layout. We use 180 nm TSMC technology as the example. Each PMOS transistor was 540 x 180 nm. Each NMOS transistor was 270 x 180 nm. There were 7 pins: *Dtrain*, *Vin*, *E*, *Vout*, *Vout*, *Vdd*, and *GND*. In order to minimize the size of layout, we divided the whole design into two parts but sharing the same GND. In the layout, the part on the bottom was upside down to share the GND metal.

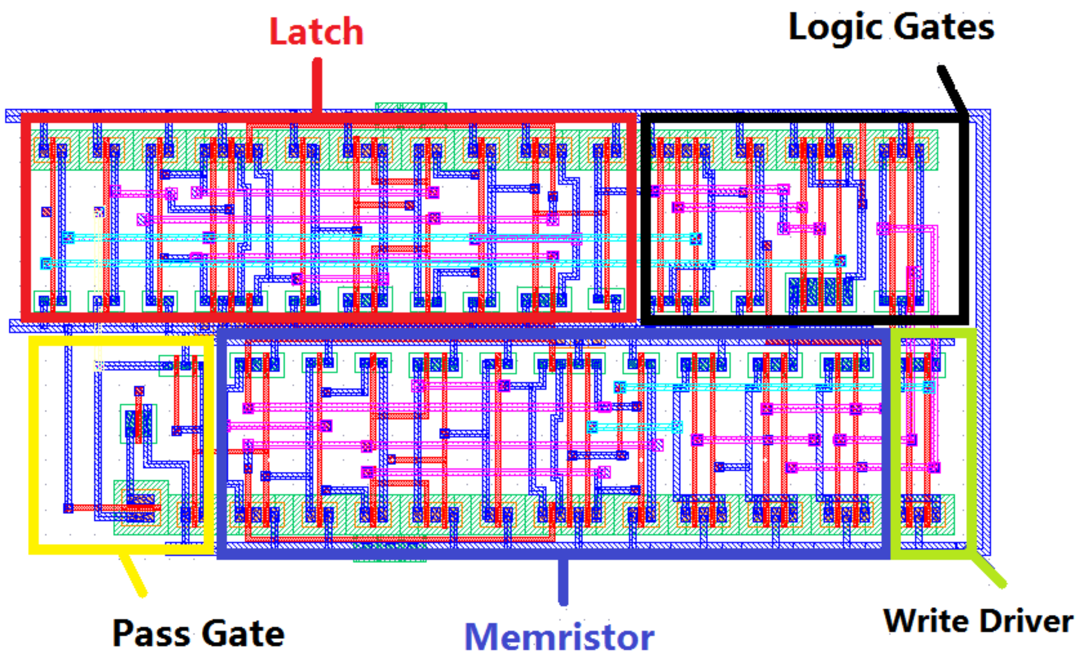
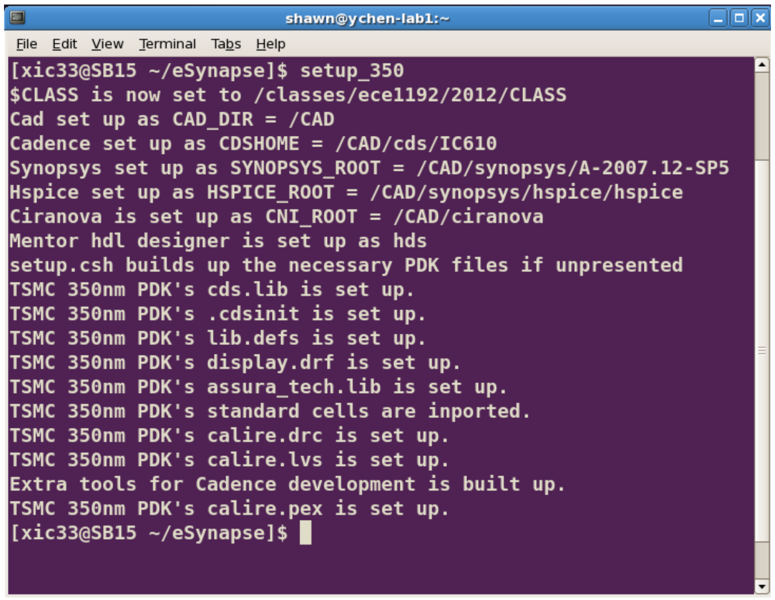


Figure 6. Layout of synapse design with training circuit

3.2.3 Design environment. We setup the design environment of the targeted tapeout process, including the library setup, and the parasitic parameter extraction flow based on Calibre (Mentor Graphics) and Assura (Cadence). We also installed the required file for chip tape-out process, e.g., TSMC 350nm technologies. All required libraries, including verification tools, were fully installed. In the installation process, the work load mainly included how to integrate the PDK into the pre-installed Cadence environment. The TSMC PDK provided the GUI symbol in Virtuoso, SPICE model, and configuration protocols for each standard cell. Also, TSMC provide both Assura and Calibre tool sets, which conduct multiple functions like Design Rule Check (DRC), Layout vs. Schematic (LVS), and RC extraction for post-layout verification. By applying these rules correctly during the test, the most significant work was to ensure the design was correct and satisfied the process and manufacture requirements. Significant effort was spent on the analysis of TSMC PDK structure, setting up the environment variables, and loading necessary configuration files. Despite the services provided in TSMC PDK, we also integrated some pre-exited tools and functions in this development environment which accommodated our developers' working habits and raised the efficiency. After the configuration, an integrated Linux script was written to setup all the environment variables and load the necessary configuration files by using only one command. Users in the same development group could simply source the script files and achieve the one-click setup at any computers.

The library installation example is shown in Figure 7. The hierarchy of installation path is summarized as below:



```
[xic33@SB15 ~/eSynapse]$ setup_350
$CLASS is now set to /classes/ece1192/2012/CLASS
Cad set up as CAD_DIR = /CAD
Cadence set up as CDSHOME = /CAD/cds/IC610
Synopsys set up as SYNOPSYS_ROOT = /CAD/synopsys/A-2007.12-SP5
Hspice set up as HSPICE_ROOT = /CAD/synopsys/hspice/hspice
Ciranova is set up as CNI_ROOT = /CAD/ciranova
Mentor hdl designer is set up as hds
setup.csh builds up the necessary PDK files if unpresented
TSMC 350nm PDK's cds.lib is set up.
TSMC 350nm PDK's .cdsinit is set up.
TSMC 350nm PDK's lib.defs is set up.
TSMC 350nm PDK's display.drf is set up.
TSMC 350nm PDK's assura_tech.lib is set up.
TSMC 350nm PDK's standard cells are inported.
TSMC 350nm PDK's calire.drc is set up.
TSMC 350nm PDK's calire.lvs is set up.
Extra tools for Cadence development is built up.
TSMC 350nm PDK's calire.pex is set up.
[xic33@SB15 ~/eSynapse]$
```

Figure 7. Tapeout library installation

```
|- assura_tech.lib  cds.lib  display.drf  techfile  icc.rrules ...
|- pdkInstall.cfg  pdkInstall.pl  Readme.first  REVISION
|- /setup
    |- cdsenv  cdsinit  cds.lib  common_bindkeys.il
    |- runset.calibre.drc runset.calibre.pex  runset.calibre.lvs
    |- setup_TSMC350
|- /skill
    |- callback.ile  pdkParamTable.ile ...
|- /models
    |- tsmc35lg.scs  tsmc35lg.mdl  CM_FSG.mdle  LP_FSG.mdl  LV_FSG.mdl
|- /stream
    |- strmioMap
|- /techFiles
    |- /Assura
        |- drc  lvs ...
    |- /Calibre
        |- drc lvs ...
    |- icc.rules  cds.lib  ...
|- /Assura
    |- /drc - assura.ant  assura.drc ...
    |- /lvs - extract.rul   bind.rul  compare.rul ...
    |- DRC.README  LVS.README  techRuleSets ...
|- /Calibre
    |- /drc - calibre.ant  calibre.drc
    |- /lvs -  calibre.lvs  calibre.xrc ...
    |- DRC.README  LVS.README  REVISION.calibre
|- /PDK_doc
|- Device_list.txt  Application_note_for _customized_cells.pdf  tsmc_PDK_usage_guide.pdf ...
```

3.3. Synapse Neuromorphic System Design

3.3.1 Biology-inspired synapse design. In biological neural systems, there are chemical synapses and electrical synapses working as the connections between neurons. Electrical synapses can have the similar bidirectional transmission, which is important to the implementation of artificial neuromorphic computing system. In order to realize bidirectional transmission, two sets of input/output are needed for one synapse; and the transmission direction should be controlled by a switch signal. Figure 8 shows the schematic of a memristor-based bidirectional transmission synapse with two connected neurons. The resistance *R* and memristance *M* determined the gate voltage of the NMOS transistor. By controlling the gate voltage of the NMOS transistor, the weighted current was generated just like biology synapse. To test the bidirectional transmission function of the synapse, the neurons worked as an oscillating ring. Each neuron updated the state of the other neuron based on its own state. A neuron consisted of a capacitor and an inverter. The capacitor allowed the neuron to collect information, i.e., weighted current, from multiple inputs; and the inverter worked as an analog amplifier whose threshold determined the state of neuron according to the accumulated charge on the capacitance.

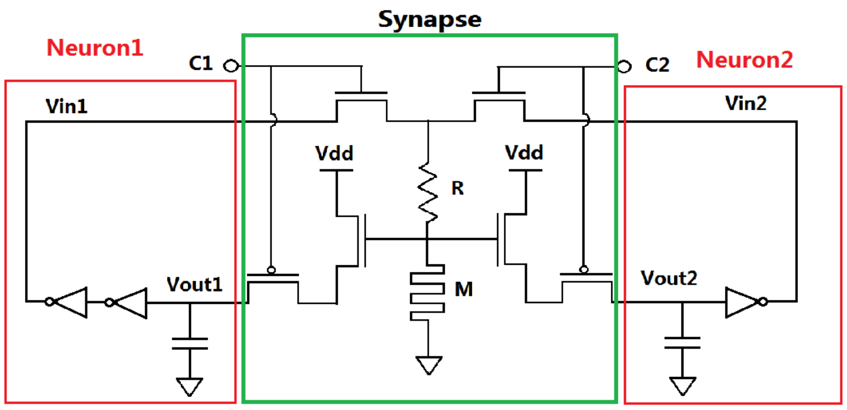


Figure 8. Schematic of neurons with memristor-based bidirectional synapse

In biological neural systems and artificial neural network models, synapses transmit excitations or inhibitions between neurons according to different functions. Inhibitions are required in the system to suppress the excitations in interconnected networks. Interneurons, by way of their inhibitory actions, provide the necessary autonomy and independence to neighboring principal cells. Therefore, one of the basic components of neuromorphic computing systems is synapses with the ability of excitation/inhibition transmission. However, in the literature, prior research on memristor-based synapse design only focused on weighted excitation signal transmission. Figure 9 shows the schematic of the synapse we proposed to implement excitation/inhibition transmission. Since a capacitor was used in the neuron design, excitation/inhibition could be implemented by charging (pull up)/discharging (pull down) the capacitance. The truth table of the Excitation/Inhibition synapse is shown in Table 3. Signal '+/-' was used to determine whether a synapse implemented an 'Excitation' or 'Inhibition' function. When Vin1/Vin2 was '1' and '+/-' was '1', the input signal was positive. The synapse then transmitted an excitation. Next, *Vc* (XOR output of input signal '*Vin1*/*Vin2*' and '+/-') became '0', which enabled the P-transistor and cut off the N-transistor, charging the neuron connected to it.

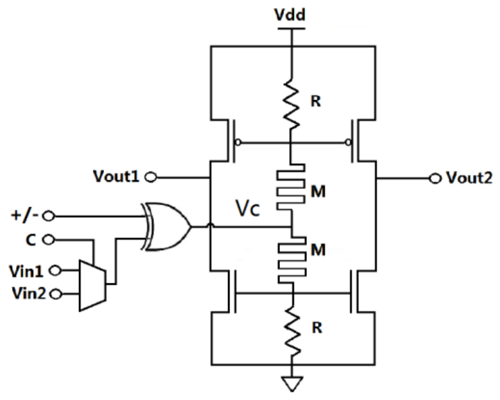


Figure 9. Excitation/Inhibition synapse

Table 3. Excitation/Inhibition synapse truth table

Vin1/Vin2	+/–	P-transistor	N-transistor	Vout1/Vout2
1	1	Pass	Cut off	Pull up
1	0	Cut off	Pass	Pull down
0	1	Cut off	Pass	Pull down
0	0	Pass	Cut off	Pull up

To demonstrate the weighted Excitation/Inhibition transmission ability of the synapse we proposed, simple information collecting neuron demo (shown in Figure 10) was designed and simulated in the Cadence environment.

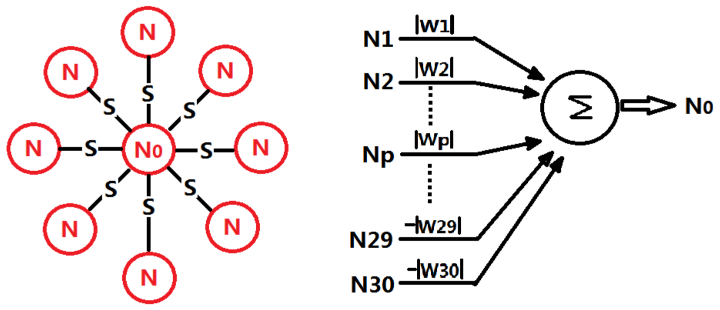


Figure 10. Information collecting neuron demo

In this demo, neuron *N0* collected the information from other neurons through synapses as we proposed. Based on the weighted Excitation/Inhibition signals from 30 other neurons, neuron *N0* make a decision with a non-linear function

$$N0 = \begin{cases} 1 & if \sum_{i=0}^{30} N_i \times W_i \geq threshold \\ 0 & otherwise \end{cases} \tag{16}$$

3.3.2 Synapse based neuromorphic computing system. A memristor behaves similarly to a synapse in biological systems and hence can be easily used as weighted connections in neural networks. Based on the memristor-based bidirectional synapse design, we implemented a network serving as a neuromorphic computing system with units (artificial neurons) and weighted connections (synapses). The neuron in this network was a binary threshold unit that produced only two different states (values). A synapse worked as a weighted connection to transmit a signal from one neuron to another. The activation function was described in equation (14).

The proposed neural network could be used for pattern recognition. Frst, multiple standard input images were sent in to train the connection weights of the system till they converged. After that, every input pattern produced a local minimum, which was a stable state corresponding to one of the stored standard patterns. Such a network system was then used to recognize the input image with defects. In our experiment, we built a network with 100 (10×10) neurons and stored the character images ‘A’, ‘B’, and ‘C’ shown in Figure 11(a) as the standard patterns. Each neuron in the network represented a pixel of the image. Then the defected images in Figure 11(b) were applied as the inputs to initialize the network’s state. Each input had 13 defects compared to its corresponding standard images (see black bars), as shown in Figure 11(b). The proposed system successfully recognized the imperfect images and converged to one of the standard patterns, as demonstrated by the write bars in Figure 11(c).

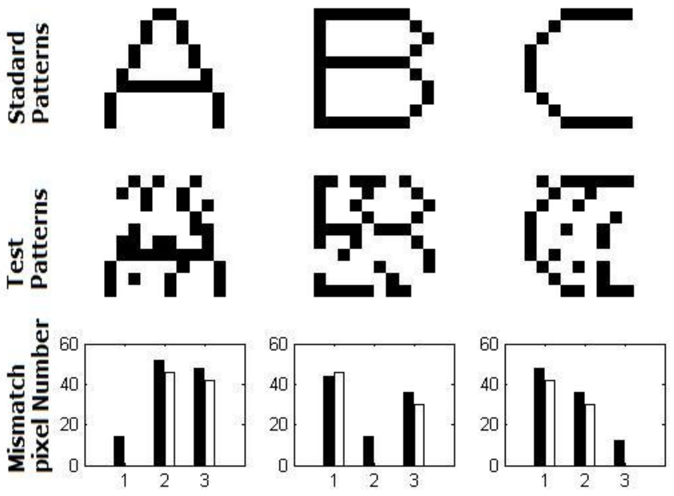


Figure 11. The neural network for pattern recognition: (a) the standard patterns and (b) the noised input patterns. (c) Comparison of the convergence iterations when recognizing the noisy images input (black bars) and the standard images (white bars)

4.0 RESULTS AND DISCUSSION

In the section, we showed the corresponding simulation results to validate our proposed model and the designed memristor-based synapsed based design.

4.1. Validation of Memristor Model

Table 4 summarized the three types of the device parameters used in our memristor model, including the geometric parameters, the electrical parameters, and the structural parameters. The electron generating coefficients $\boldsymbol{\gamma_t}$ and $\boldsymbol{\gamma_i}$ were derived from the measured data and assumed constant for the different working ranges, as shown in Figure 12. We compared our model with the experimentally obtained characteristic static I-V curve and dynamic pulse programming curve of a TiO_2-TiO_{2-x} memristor device. Figure 12 shows the measured I-V curve from a real memristor device as well as the experimental data from the numerical simulation and analytical approximation. During the measurement, a sequence of voltage pulses was applied to the memristor device. The magnitude of the voltage pulse grew exponentially and varied from positive to negative following a sinusoidal function. The numerical simulation results fit well with the measured data in all four working ranges. The analytical approximation showed slight discrepancy from the measured data when the resistance was high. This was because the simulated doping front velocity was lower than the actual value when the variation of $\boldsymbol{n_i}$over time was ignored.

Table 4. Model Parameters

	Parameter	Value	Parameter	Value
Geometric	D	35 nm	A	25 μm^2
Electrical	e	1.602×10^{-19} C	n_c	8.75×10^{20} m^{-3}
Structural	w_0	$0.15D$	λ_0	$0.05D$
Working Range	Derived parameters			
1→2	γ_t	2.3×10^{-6}	γ_i	1×10^{-6}
2→1		8×10^{-6}		1×10^{-8}
1→3		1×10^{-10}		1×10^{-9}
3→1		7×10^{-7}		2×10^{-7}

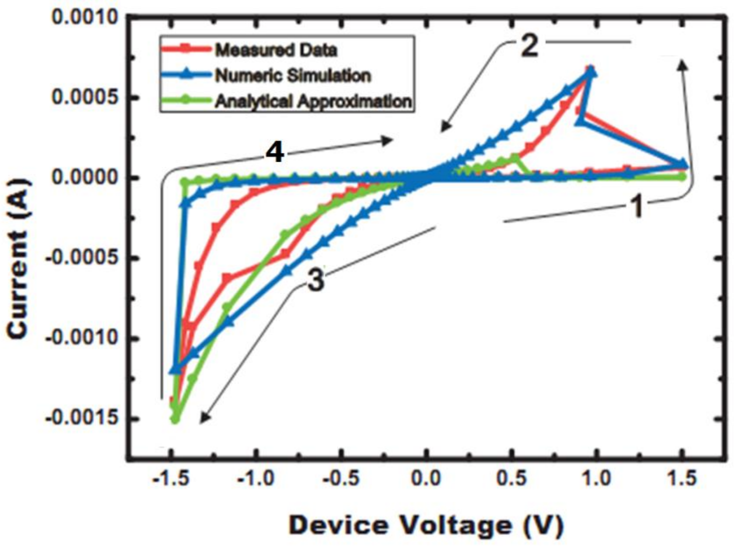


Figure 12. Model validation with static I-V curve, including numerical simulation and analytical approximation

To prove the capability of our model for simulating the dynamic switching property of the memristor, we plotted the resistance changes following the programming pulses in Figure 13. The resistance of the memristor first decreased when the positive pulses were applied and then rose when the polarization of the pulses changed to negative. Our numerical simulation matched the measured data very well over most of the plotted points. Small discrepancies appeared at the high resistance state. One reason for the deviations could be the impact of thermal fluctuations, which become prominent under a relatively low programming voltage. The analytical approximation showed relatively large deviation from the measured data at the high resistance state.

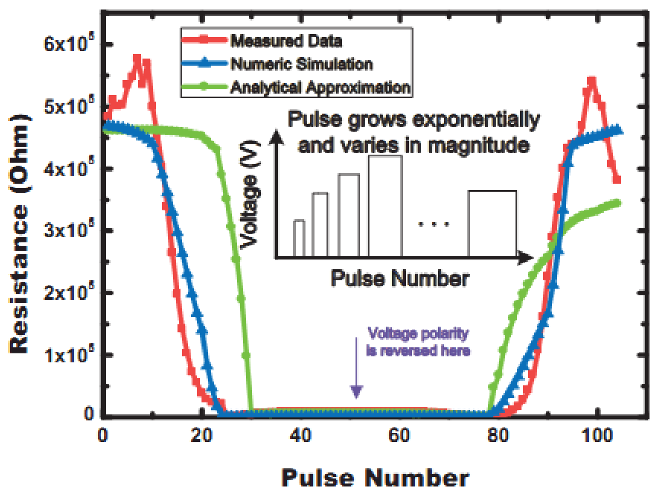


Figure 13. Model validation of memristor dynamic switching for one single cycle, including numerical simulation and analytical approximation

4.2. Simulations of the CMOS-mimicked Memristor Training Circuit

4.2.1 Pre-layout results. The pre-layout simulation results of the CMOS-mimicked memristor training circuit are shown in Figure 14. The *Vout* (*vout* in the schematic) was stored in the latch when the signal *E* rose. *Vout* remained constant during the training period. The memristor was trained when *E* rose to high. This caused a delay in *Vout* due to the training time. When *E* was pulled down, *Vout* changed according to the resistance of the memristor (or NMOS transistor). From Figure 14, the CMOS-mimicked memristor was trained only when the *Vout* was different from the *D_train* (*Dtrain* in the schematic).

4.2.2 Post-layout results. The simulation results based on the extracted layout parameters are shown in Figure 15. The circuit worked as expected. This simulation verified that our CMOS-mimicked memristor-based synapse design fully demonstrated the required functionalities of the biological synapses. The training circuit worked well at the circuit level.

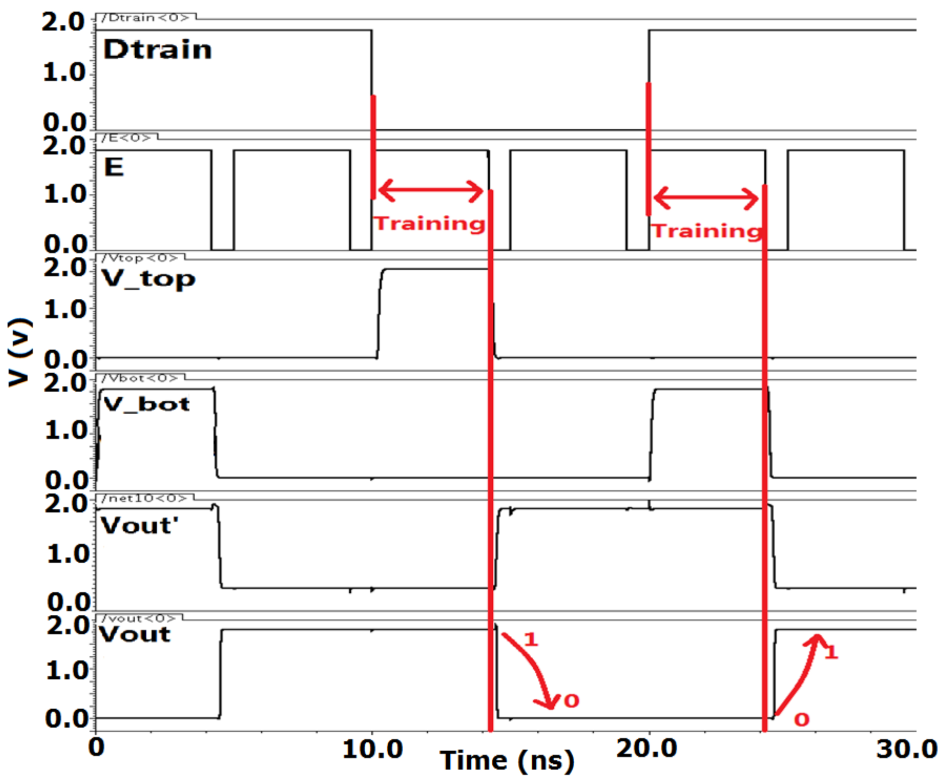


Figure 14. Simulation of synapse circuit schematic, including training circuit

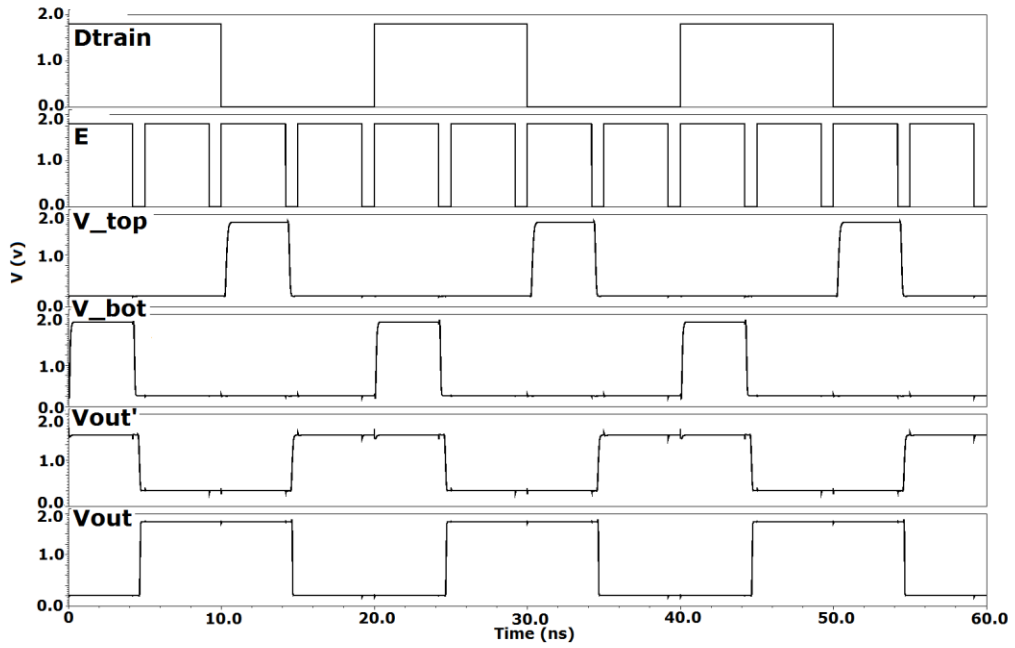


Figure 15. Simulation of synapse circuit on extracted layout

4.3. Simulations of Memristor-based Bidirectional Synapse Design

4.3.1 Results of neuron-synapse oscillating ring. The neuron-synapse oscillating ring based on the memristor-based bidirectional synapse design was simulated in Cadence Virtuoso, as shown in Figure 16. When the switch signal *C1* was '1', *neuron1* updated the state of *neuron2* to *neuron1*'s state. When *C1* was '0', the synapse worked the other way – *neuron2* changed the state of *neuron1* to the opposite state of *neuron2*. In the beginning of the simulation, the initial states of both neurons was set to '1'. At 30 µs, the capacitance of *neuron1* was discharged (*Vin2* was '0'). Since the data was stored as the capacitance, the neuron oscillating ring had very good tolerance for race conditions.

An advantage of the neuron-synapse oscillating ring was that the oscillating frequency was determined by the weight of the synapse. The larger the memristance was, the higher the gate voltage was; and, consequently, the stronger the weighted current that would be generated. This meant the charging period of the capacitance was longer.

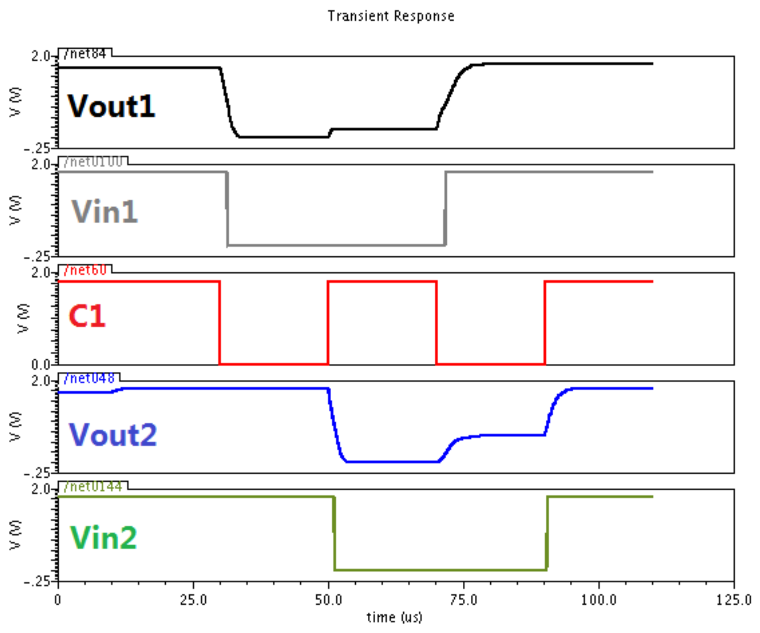


Figure 16. Simulation result of neuron-synapse oscillating ring

4.3.2 Results of information collecting neuron demo. We gave different sets of excitation and inhibition weights to every synapse in Figure 10 and tested the state of *N0* by increasing the number of positive input neurons. Result are shown as Figure 17.

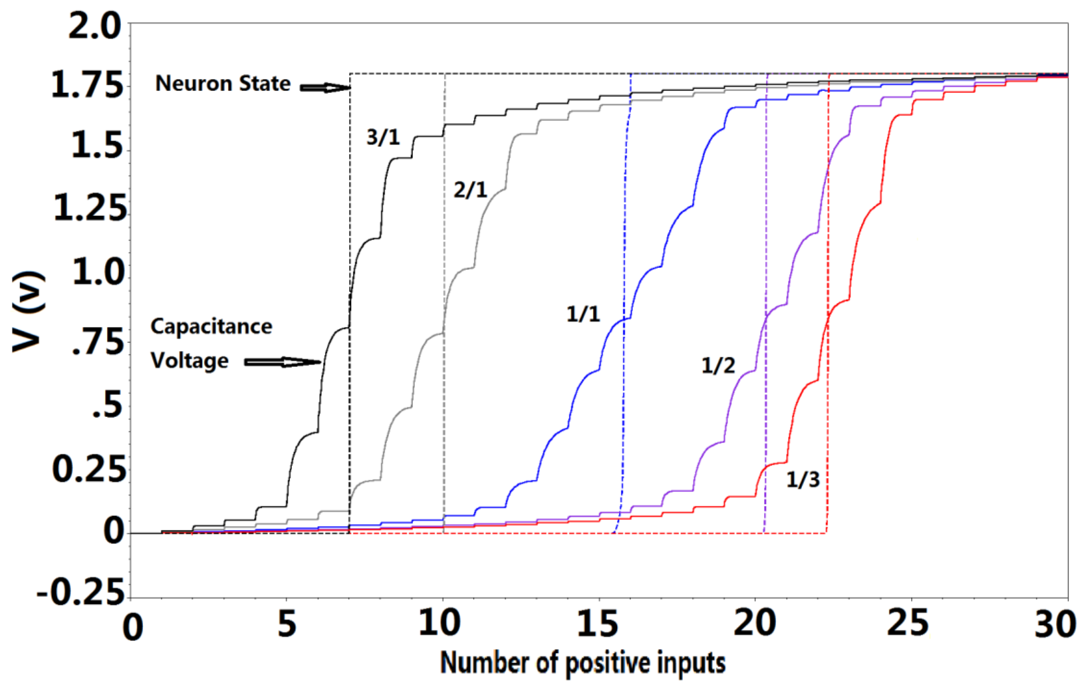


Figure 17. Functions of the synapse network with memristor-based bidirectional synapses

Curves from left to right depict the test results for positive/negative synapse weight ratios of 3/1, 2/1, 1/1, 1/2, and 1/3. The number of positive neurons needed to change the state of *N0* was 7, 10, 15, 21, and 23, respectively.

4.3.3 **Robustness.** The number of storable standard patterns (capacity) of this neural network design was determined by the number of neurons and connections. Also, the more patterns stored in the system, the higher the precision of the connection weights was needed. Therefore, a large number of stored patterns and a high process variation of memristances would result in a high failure probability Pf. To quantitatively evaluate the impact of memristance variations and the robustness of the proposed neural network design, we conducted Monte-Carlo simulations on the network with 100 (10×10) neurons. Random variations following Gaussian distribution were injected into the memristors. The system could fail to recognize the noised patterns or mismatch an input with other standard patterns due to the inaccurate connection weights. To test the failure probability under different conditions, we ran 10,000 Monte-Carlo simulations by varying the memristance variation σ, standard deviation of memristance, for 7, 8, 9, or 10 stored patterns in the system. In this experiment, each input image contains 21 defects among 100 pixels.

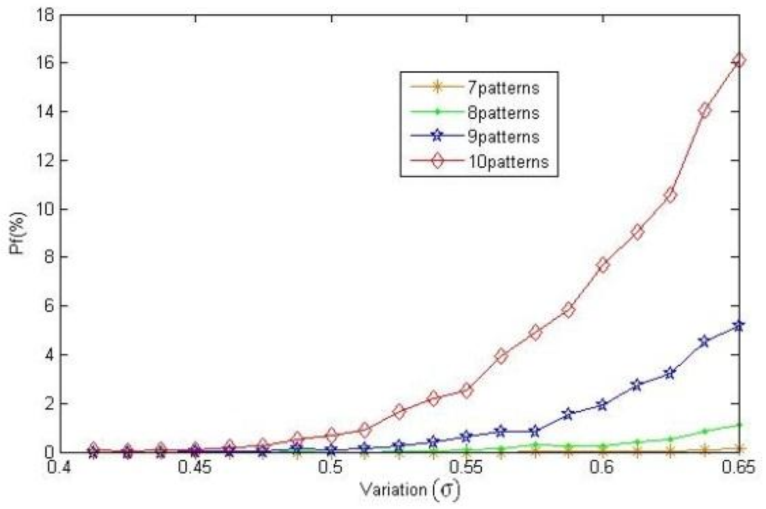


Figure 18. The impact of memristor variations on the failure probability *Pf*

The simulation results in Figure 18 demonstrated that the proposed memristor-based neuromorphic system had a high tolerance on memristance variations. When $\sigma < 0.4$, *Pf* of all the four configuration were close to the ideal condition at $\sigma = 0$. This indicated that the performance of the proposed neuromorphic system maintained robustness even with a large memristor device variation. When further increased to $\sigma > 0.5$, *Pf* grew significantly. As expected, under the same process variation condition, the system suffered from a higher *Pf* when more patterns were stored.

4.3.4 Capacity analysis. In our implemented artificial neural network, the capacitance worked as a key factor affecting the robustness of the system. If errors in recollection were allowed, the maximum number p of the patterns to be stored in a network with N neurons is $0.15N$. This limitation was attributed to the fact that the network was trapped in the so called "spurious local minima." In 1987 McEliece, et al. [8] proved that when $p < N/(4\ln N)$ held true, the Hopfield's model was able to recall all the memorized patterns without errors.

For demonstration purpose, we conducted Monte-Carlo simulations to evaluate the impacts of the capacity on the robustness of our networks. A large Hopfield network with 100 neurons was built to recognize larger sets of text patterns where the respective theoretical capacity was limited to about 18 patterns. Process variations were simulated by introducing Gaussian distribution noise to the memristance of the memristor devices in Matlab simulations. A system failure was defined as converging to a wrong standard pattern (one that does not correspond to the input pattern) or failing to converge to a stable point. The test results are shown in Figure 19.

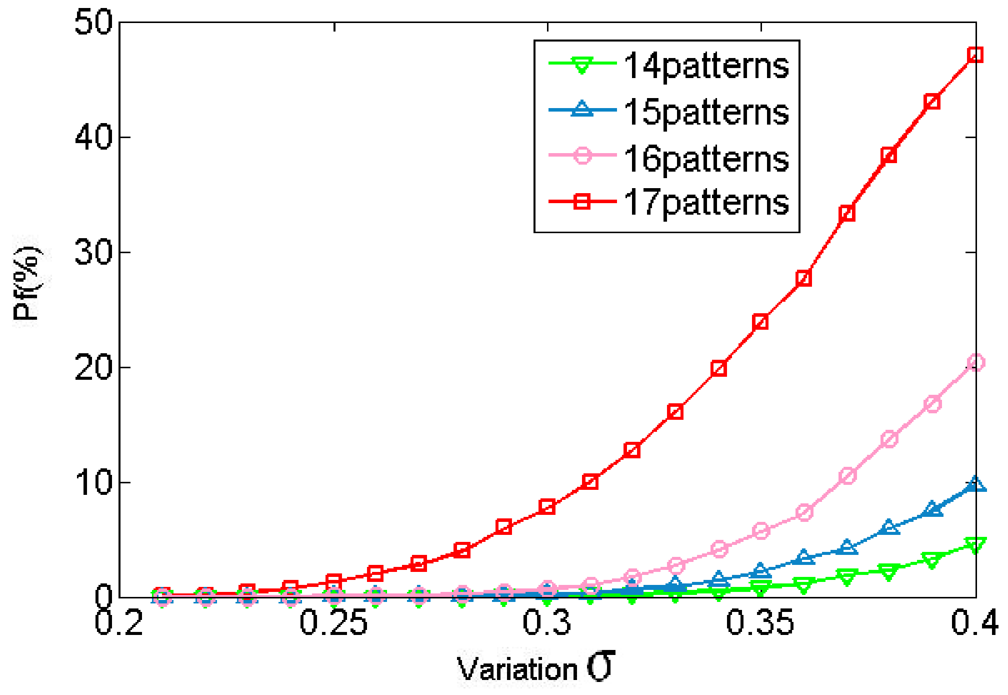


Figure 19. Failure rate of memristor-based Hopfield network under different pattern numbers and process variation conditions

Figure 19 shows that our design had a good immunity against process variations. Even when $\sigma < 0.2$, our system still demonstrated a *Pf* close to zero. Increasing the number of text patterns quickly degraded the system's robustness, i.e., a much higher *Pf*. When the number of patterns approached the capacity limit, the system robustness degraded very quickly. The increase in process variations σ also substantially degraded system robustness. However, in conventional CMOS circuit manufacturing, the parametric standard deviation is usually less than 10% [9].

For the same amount of stored patterns, a larger network with more neurons was more robust to process variations. Figure 20 compares the performance of the systems with 100 neurons (the blue line) and with 400 neurons (the green line). Both systems had 10 standard patterns, and the input defect rate remains at 21% for the two designs. The simulations show that in a bigger network, the impact of process variations was smaller, leading to a lower required precision of the connection weights. Hence, in a neural network system design, the tradeoff between network capacity and robustness needs to be considered.

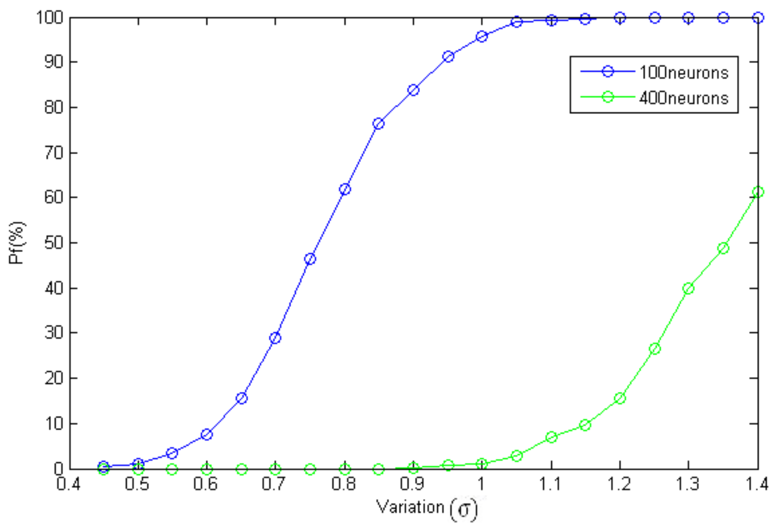


Figure 20. Increasing the network size vs. *Pf*

5.0 CONCLUSIONS

In this project, we proposed a compact model to simulate the transition region motion in the TiO_2-TiO_{2-x} memristor based on classic ion transportation theory. Our model was validated with the measured data from a real TiO_2-TiO_{2-x} memristor device and proved capable of simulating the static and dynamic switching properties of the device. We then designed a memristor-based synapse circuit with bidirectional transmission and exhibition/inhibition functions and implemented neuromorphic computing system with our proposed synapse design. Experimental results showed that the proposed design had high tolerance on process variation and input noise. Finally, we compared memristor crossbar-based and synapse-based neuromorphic computing architectures and discussed their respective advantages and target applications.

6.0 REFERENCES

[1] L. Chua, “Memristor - the Missing Circuit Element,” *IEEE Transactions on Circuit Theory*, vol. 18, pp. 507-519, 1971.

[2] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, “The Missing Memristor Found,” *Nature*, vol. 453, pp. 80-83, 2008.

[3] L. Chua, “Resistance Switching Memories are Memristors,” *Applied Physics A: Materials Science & Processing*, vol. 102, no. 4, pp. 765-783, 2011.

[4] H. Wang, H. Li, and R. Pino, “Memristor-based Synapse Design and Training Scheme for Neuromorphic Computing Architecture” *International Symposium on Circuit and Systems*, 2011.

[5] X. Wang, Y. Chen, H. Xi, H. Li, and D. Dimitrov, “Spintronic Memristor through Spin-torque-induced Magnetization Motion,” *IEEE Electron Device Letters*, vol. 30, pp. 294-297, 2009.

[6] Y. Ho, G. Huang, and P. Li, “Nonvolatile Memristor Memory: Device Characteristics and Design Implications,” *International Conference on Computer-Aided Design*, Nov 2009, pp. 485-490, 2009.

[7] D. Niu, Y. Chen, C. Xu, and Y. Xie, “Impact of Process Variations on Emerging Memristor,” *Design Automation Conference (DAC)*, pp. 877–882, 2010.

[8] R. J. McEliced, E. C. Posner, E. R. Rodemick, and S. S. Venkatesh. “The Capacity of the Hopfield Associative Memory.” *IEEE Transactions on Information Theory*, IT-33, pp. 461-482, 1987.

[9] R. E. Pino, H. Li, Y. Chen, M. Hu, and B. Liu. “Statistical Memristor Modeling and Case Study in Neuromorphic Computing” *49th Design Automation Conference*, pp. 585-590, 2012.

Publications and Presentations

1. Lu Zhang, Zhijie Chen, J. Joshua Yang, Bryant Wysocki, Nathan McDonald and Yiran Chen, "A Compact Modeling of TiO_2-TiO_{2-x} Memristor", *to be submitted*.

2. B. Liu, Y. Chen, B. Wysocki, and T. Huang, "The Circuit Realization of a Neuromorphic Computing System with Memristor-based Synapse Design," *International Conference on Neural Information Processing*, Nov 2012, pp. 357-365, 2012.

LIST OF ABBREVIATIONS AND ACRONYMS

CMOS	complementary metal–oxide–semiconductor
DRC	Design Rule Check
HP	Hewlett-Packard
LVS	Layout vs. Schematic
MOSFET	metal–oxide–semiconductor field-effect transistor
NMOS	N-channel MOSFET
Pf	probability of failure
PMOS	P-channel MOSFET
RC	resistor-capacitor
SPICE	Simulation Program with Integrated Circuit Emphasis